A PBL Report

On

Spam Email Detection Using Machine Learning

submitted to CMREC (UGC Autonomous)

In Partial Fulfillment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

(Artificial Intelligence and Machine Learning)

Submitted By

B. Bala Bhaskar - 208R1A05C7

CH. Nithish Kumar - 208R1A05D3

P.R. Dhanush Reddy -208R1A05G8

T. Rohith reddy –208R1A05H5

Under the guidance of

Mrs. G. Sumalatha

Assistant Professor, Department of CSE

Department of Computer Science & Engineering

CMR ENGINEERING COLLEGE

(Accredited by NBA, Approved by AICTE, NEW DELHI, Affiliated to JNTU, Hyderabad)

Kandlakoya,Medchal Road, R.R. Dist. Hyderabad-501 401)

(2022-2023)

CMR ENGINEERING COLLEGE

(Accredited by NBA,Approved by AICTE NEW DELHI, Affiliated to JNTU, Hyderabad)
Kandlakoya, Medchal Road, Hyderabad-501 401

Department of Computer Science & Engineering

CERTIFICATE

This is to certify that the project entitled

"RANDOM PASSWORD GENERSTOR"

is a bonafide work carried out by

B. Bala Bhaskar-208R1A05C7
CH. Nithish Kumar– 208R1A05D3
P.R. Dhanush Reddy-208R1A05G8
T. Rohith reddy – 208R1A05H5

in partial fulfillment of the requirement for the award of the degree of **BACHELOR OF TECHNOLOGY** in **COMPUTER SCIENCE AND ENGINEERING** from CMR Engineering College, under our guidance and supervision.The results presented in this project have been verified and are found to be satisfactory. The results embodied in this project have not been submitted to any other university for the award of any other degree or diploma.

Internal Guide	Head of the Department
Mrs. S.Sumalatha	**Dr.Sheo kumar**
Assistant Professor	Professor & HOD
Department of CSE ,	Department of CSE ,
CMREC, Hyderabad	CMREC, Hyderabad

DECLARATION

This is to certify that the work reported in the present project entitle

"SPAM EMAIL DETECTION USING MACHINE LEARNING"

is a record of bonafide work done by me in the Department of Computer Science and Engineering , CMR Engineering College. The reports are based on the project work done entirely by me and not copied from any other source. I submit my project for further development by any interested students who share similar interests to improve the project in the future.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

B. Bala Bhaskar-208R1A05C7
CH. Nithish Kumar– 208R1A05D3
P.R. Dhanush Reddy-208R1A05G8
T. Rohith reddy – 208R1A05H5

SPAM EMAIL DETECTION USING MACHINE LEARNING

Introduction:

In the era of information technology, information sharing has become very easy and fast. Many platforms are available for users to share information anywhere across the world. Among all information sharing mediums, email is the simplest, cheapest, and the most rapid method of information sharing worldwide. But, due to their simplicity, emails are vulnerable to different kinds of attacks, and the most common and dangerous one is spam [1]. No one wants to receive emails not related to their interest because they waste receivers' time and resources. Besides, these emails can have malicious content hidden in the form of attachments or URLs that may lead to the host system's security breaches [2]. Spam is any irrelevant and unwanted message or email sent by the attacker to a significant number of recipients by using emails or any other medium of information sharing [3]. So, it requires an immense demand for the securityof the email system. Spam emails may carryviruses, rats, andTrojans. Attackers mostly usethis technique for luring users towards online services. They may send spam emailsthat contain attachments with the multiple-file extension, packed URLs that lead the user to malicious and spamming websites and end up with some sort of data or financial fraud and identify theft [4, 5]. Many email providers allow their users to make keywords base rules that automatically filter emails. Still, this approach is not very useful because it is difficult, and users do not want to customize their emails, due to which spammers attack their email accounts.

Social networking experts estimate that 40% of social network accounts are used for spam [8]. The spammers use popular social networking tools to target specific segments, review pages, or fan pages to send hidden links in the text to pornographic or other product sites designed to sell something from fraudulent accounts. The noxious emails that are sent to the same kind of individuals or associations share regular highlights. By investigating these highlights, one can improve the detection of these types of emails. By utilizing artificial ntelligence (AI) [9], we can classify emails into spam and nonspam emails. This solution is possible by using feature extraction from the messages' headers, subject, and body. After extracting this data based on their nature, we can group them into spam or ham. Today, learning-based classifiers [10] are commonly used for spam detection. In learning-based classification, the detection process assumes that spam emails have a specific set of features that differentiate them fromlegitimate emails.

One example of learning-based models is extreme learning machine (ELM). This is a modern machine learning model for the feedforward neural networks containing only one hidden layer [12]. It eliminates slow training speed and overfitting problems when compared with traditional neural networks. In ELM, it requires only one cycle of iteration. Because of better generalization potential, robustness, and controllability, this algorithm specifically is now used in many fields. In this paper, we consider different machine learning algorithms for spam detection. Our contributions are delineated as follows:(i)The study discusses various machine learning- based spam filters, their architecture, along with their pros and cons. We

alsodiscussed the basic features of spam email.(ii)Some exciting research gaps were found in the spam detection and filtering domain by conducting a comprehensive survey of the proposed techniques and spam's nature.(iii)Open research problems and future research directions are discussed to enhance email security and filtration of spam emails by using machine learning methods.

Problem Statement:

The increasing volume of spam emails has become a significant challenge in today's digital world. These emails can contain malicious links, viruses, or phishing attacks that can compromise user security and privacy. Therefore, there is a need for an effective spam email detection system using machine learning techniques. The goal of this project is to develop a machine learning model that can accurately classify emails as spam or non- spam. The system should be able to analyze the email's content, sender, and other relevant features to determine its spam status. The model should also be able to adapt to new types of spam emails and maintain a low false- positive rate. The proposed system can help users filter out spam emails and improve their overall email security and privacy.

Existing Systems:

There are several existing systems for spam email detection using machine learning. Here are some examples:

SpamAssassin: SpamAssassin is an open-source spam filter that uses machine learning algorithms to detect spam emails. It uses a set of rules to assign a score to each email, and if the score exceeds a certain threshold, the email is marked as spam.

Gmail: Google's email service, Gmail, also uses machine learning algorithms to detect and filter spam emails. It analyzes various features of an email, such as the sender's address, content, and metadata, to determine whether it is spam.

Microsoft Exchange: Microsoft Exchange is an email server that includes built-in spam filtering using machine learning algorithms. It uses a variety of techniques, including content filtering and reputation analysis, to detect and filter spam emails.

Apache OpenNLP: Apache OpenNLP is an open-source machine learning library that can be used for text analysis tasks such as spam detection. It provides tools for natural language processing and text classification, which can be used to train models for detecting spam emails.
TensorFlow: TensorFlow is an open-source machine learning framework that can be used for a variety of tasks, including spam detection. It provides tools for building and training deep neural networks, which can be used to analyze the content of emails and determine whether they are spam.

These are just a few examples of the many existing systems for spam email detection using machine learning. The specific approach used by each system may vary, but the general idea is to use machine learning algorithms to analyze various features of an email and determine whether it is likely to be spam.

Proposed Systems:

There are several proposed systems in spam email detection using machine learning. Here are some of them:

Rule-based classification: This system involves setting up a set of rules based on various features of spam emails, such as the presence of certain keywords, the use of all caps, excessive punctuation, or multiple exclamation marks. These rules are then used to classify incoming emails as spam or not spam.

Bayesian filtering: This system uses statistical methods to analyze the content of incoming emails and assign probabilities to them being spam or not spam. This is done by analyzing the frequency of certain words and phrases in spam emails and non-spam emails and using this information to make predictions about new emails.

Support Vector Machines (SVM): This is a popular machine learning algorithm that is used in spam email detection. It involves training a classifier on a dataset of labeled emails (spam or not spam), and then using this classifier to classify new emails based on their content.

Naive Bayes Classifier: This is another machine learning algorithm commonly used in spam email detection. It works by using Bayes' theorem to calculate the probability that an email is spam based on the presence of certain words or phrases in the email.

Artificial Neural Networks (ANN): This is a type of machine learning algorithm that mimics the structure and function of the human brain. It can be used to classify emails as spam or not spam based on their content.

These are just a few of the proposed systems in spam email detection using machine learning. Depending on the specific needs of a given organization, different approaches may be more effective than others..

Techniques / Algorithms:

Spam email detection is a classic example of a binary classification problem where the task is to determine whether an email is spam or not spam (ham). Machine learning techniques can be used to develop models that can accurately classify emails as spam or ham based on their content and other features.

Here are some popular techniques and algorithms used in spam email detection:

Naive Bayes Classifier: Naive Bayes is a probabilistic algorithm that is widely used for text classification tasks. It works by calculating the probability of each word in an email belonging to a particular class (spam or ham) and then combining these probabilities to give an overall classification. Naive Bayes is known for its simplicity and efficiency, making it a popular choice for spam email detection.

Support Vector Machines (SVMs): SVMs are a powerful algorithm that can be used for binary classification tasks. They work by finding a hyperplane that separates the two classes (spam and ham) with the largest possible margin. SVMs are known for their ability to handle high-dimensional data and their robustness to noisy data.

Decision Trees: Decision trees are a simple yet powerful algorithm that can be used for binary classification tasks. They work by recursively splitting the data into subsets based on the value of a particular feature. Each split is chosen to maximize the information gain, which measures how much the split reduces the uncertainty in the classification.

The methodology is used for the method of e-mail spam filtering based on Naïve Bayes algorithm.

A. Data Preprocessing

Data Preprocessing is a strategy that is used to transform the raw information into a clean data set. In other words, whenever the information is gathered from different sources it's collected in raw format which isn't feasible for the analysis. This involves the consecutive steps:

Tokenization: Tokenization is claimed to be dividing an outsized quantity of text into smaller chunks referred to as Tokens. These tokens are pretty useful to search out the patterns and that they are parted by whitespaces characters like line break, space or by punctuation characters.

Dropping Values: Dropping is the most common method to take care of the missed values. Those rows in the data set or the entire columns with missed values are dropped in order to avoid errors to occur in data analysis.

Stop Words: Stop words are English words which don't add much content to a sentence. They will safely be ignored without forgoing the meaning of the sentence.

Bag of Words: A bag-of-words is a representation of text that describes the occurrence of words within a document and it is used for extracting features from the documents. This Algorithm contains the following steps:

a. **Step 1:** Consider a random email from the spam dataset for execution.

b. ***Step 2:*** The considered email is in basic form. To perform the feature extraction/selection and classification procedure, email is required to pre-process initially.

c. ***Step 3:*** Initially, tokenize the e-mail into individual keywords. Tokenization split each individual If the duplicate values are present within the dataset, then it'll drop the duplicate values Remove the stop words from the obtained tokens.

Now we will convert the group of text into a matrix of token
counts Splitting the dataset into training data and test data.

d. ***Step 4:*** By evaluating the model on the training and testing dataset it predicts the accuracy of the model.

B. Naïve Bayes Classifier

Naïve Bayes is one of the algorithms in machine learning which implies it predicts on the basis of probability of an object. It is mainly used in text classification. It can be used for classifying spam emails as word probability plays main role here. If there's any word which occurs frequently in spam but not in ham, then that email is spam. This algorithm has become a best technique for spam detection. The Naive Bayes calculates the probability of each class and maximum probability is then chosen as an output. Naïve Bayes always provide an accurate result. The Formula for Naïve Bayes algorithm is represented as follow

$P(A|B) = P(B|A) * P(A) / P(B)$

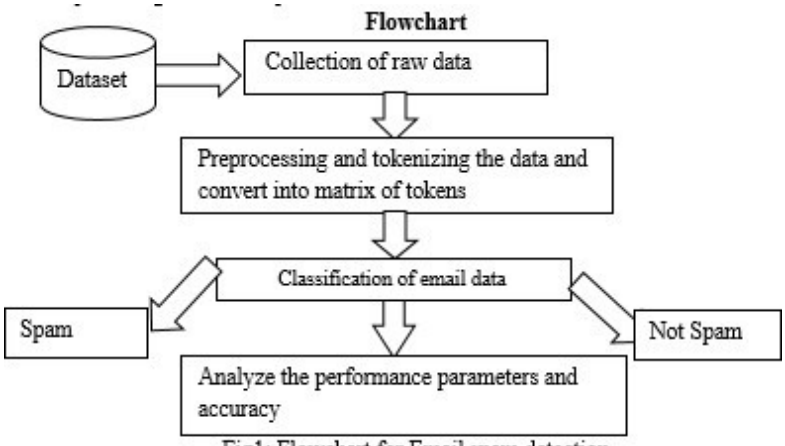


Fig1: Flowchart for Email spam detection

Implementation:

The implementation of a spam email detection system using machine learning involves the following steps:

Data collection: Collecting a dataset of emails that are labeled as spam or ham is the first step. This dataset will be used to train and evaluate the machine learning models.

Data preprocessing: The emails need to be preprocessed to convert them into a format that can be used by the machine learning algorithms. This includes removing stop words, stemming, and converting the text into numerical features using techniques such as bag-of-words or term frequency-inverse document frequency (TF- IDF).

Feature extraction: Extracting useful features from the email content is an important step in the process. This involves identifying the features that are most relevant for the classification task. Some examples of useful features for spam email detection include the sender's email address, subject line, and the frequency of certain words or phrases in the email content.

Model training: The next step is to train a machine learning model on the preprocessed and feature extracted data. The model should be chosen based on the problem requirements, dataset size, and the desired level of accuracy. The model can be trained using algorithms such as Naive Bayes, SVMs, decision trees, or neural networks.

Model evaluation: Once the model is trained, it needs to be evaluated on a test dataset to measure its performance. This involves measuring metrics such as accuracy, precision, recall, and F1-score.

Model deployment: Finally, the trained model can be deployed to classify new incoming emails as ham. This involves integrating the model into an email client or server to automatically classify incoming emails.

Overall, the implementation of a spam email detection system using machine learning requires careful consideration of the data, feature selection, model selection, and evaluation. It is important to continuously monitor the performance of the system and update it as new spam email techniques emerge.

Problem-Solving Approach:

```
# import libraries for reading data, exploring

and plotting import numpy as np import pandas

as pd import seaborn as sns import

matplotlib.pyplot as plt

from wordcloud import WordCloud, STOPWORDS, ImageColorGenerator


%matplotlib inline


# library for train test split

from sklearn.model_selection import train_test_split


# deep learning libraries for text pre-

processing import tensorflow as tf
```

```
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences


# Modeling
from tensorflow.keras.callbacks import
EarlyStopping from tensorflow.keras.models
import Sequential
from tensorflow.keras.layers import Embedding, GlobalAveragePooling1D, Dense, Dropout,
LSTM, Bidirectional
```

Let's load and explore the data now!

```
path = '/content/drive/MyDrive/spam/SMSSpamCollection'
messages = pd.read_csv(path, sep ='\t',names=["label", "message"])
messages[:3]
```

We will get the 1st three rows of our dataset in the output, which will ensure that we have properly loaded our data!

	label	message
0	ham	Go until jurong point, crazy.. Available only ...
1	ham	Ok lar... Joking wif u oni...
2	spam	Free entry in 2 a wkly comp to win FA Cup fina...

Let us get into some statistical analysis and visualize the data.

We will get short summary details of our data with the .describe() function.

```
messages.describe()
```

Let's look at the output.

	label	message
count	5572	5572
unique	2	5169
top	ham	Sorry, I'll call later
freq	4825	30

The output shows that our message count is 5572, where 2 unique labels are “spam” and “ham” , and the unique message count is 5169, as the rest are repeated ones. The top label is ‘ham’ and the top message is ‘Sorry, I’ll call later.

Since there are duplicate messages, let’s store them on a separate variable ‘duplicatedRow’ and check if it’s properly filtered out.

```
duplicatedRow = messages[messages.duplicated()]
print(duplicatedRow[:5])
```

Look at the output. It successfully printed the results.

	label	message
103	ham	As per your request 'Melle Melle (Oru Minnamin...
154	ham	As per your request 'Melle Melle (Oru Minnamin...
207	ham	As I entered my cabin my PA said, '' Happy B'd...
223	ham	Sorry, I'll call later
326	ham	No calls..messages..missed calls

	label	ham	spam
message	count	4825	747
	unique	4516	653
	top	Sorry, I'll call later	Please call our customer service representativ...
	freq	30	4

The LSTM spam detection model:

#LSTM Spam detection architecture model1 = Sequential()

model1.add(Embedding(vocab_size, embeding_dim,

input_length=max_len)) model1.add(LSTM(n_lstm,

dropout=drop_lstm, return_sequences=True))

model1.add(LSTM(n_lstm, dropout=drop_lstm,

return_sequences=True)) model1.add(Dense(1,

activation='sigmoid'))

Now, we will compile our model.

model1.compile(loss = 'binary_crossentropy', optimizer = 'adam',

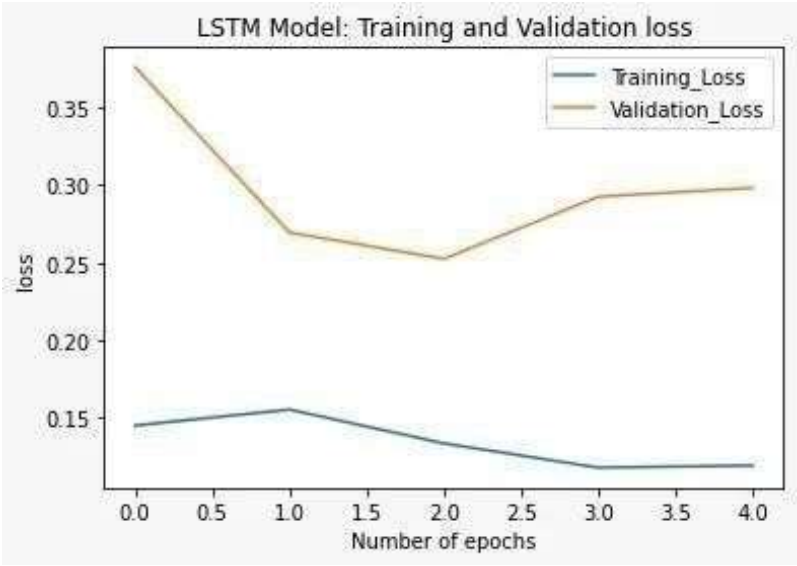
metrics=['accuracy']) Let us train our model!

num_epochs = 30 early_stop = EarlyStopping(monitor='val_loss', patience=2)

history = model1.fit(training_padded, train_labels, epochs=num_epochs, validation_data=(testing_padded, test_labels),callbacks =[early_stop], verbose=2)

```
Epoch 1/30
38/38 - 0s - loss: 0.1450 - accuracy: 0.9538 - val_loss: 0.3759 - val_accuracy: 0.8917
Epoch 2/30
38/38 - 0s - loss: 0.1553 - accuracy: 0.9487 - val_loss: 0.2694 - val_accuracy: 0.9231
Epoch 3/30
38/38 - 0s - loss: 0.1335 - accuracy: 0.9559 - val_loss: 0.2524 - val_accuracy: 0.9245
Epoch 4/30
38/38 - 0s - loss: 0.1179 - accuracy: 0.9609 - val_loss: 0.2924 - val_accuracy: 0.9240
Epoch 5/30
38/38 - 0s - loss: 0.1192 - accuracy: 0.9604 - val_loss: 0.2982 - val_accuracy: 0.9221
```

And, here is our graph..



Bi-directional Long Short Term Memory (BiLSTM) Model:

Biderectional LSTM Spam detection architecture model2 = Sequential()

model2.add(Embedding(vocab_size, embeding_dim, input_length=max_len)) model2.add(Bidirectional(LSTM(n_lstm, dropout=drop_lstm, return_sequences=True))) model2.add(Dense(1, activation='sigmoid'))

Compile, compile, compile!

model2.compile(loss = 'binary_crossentropy', optimizer = 'adam', metrics=['accuracy'])

Let’s train this smart kid.

Training

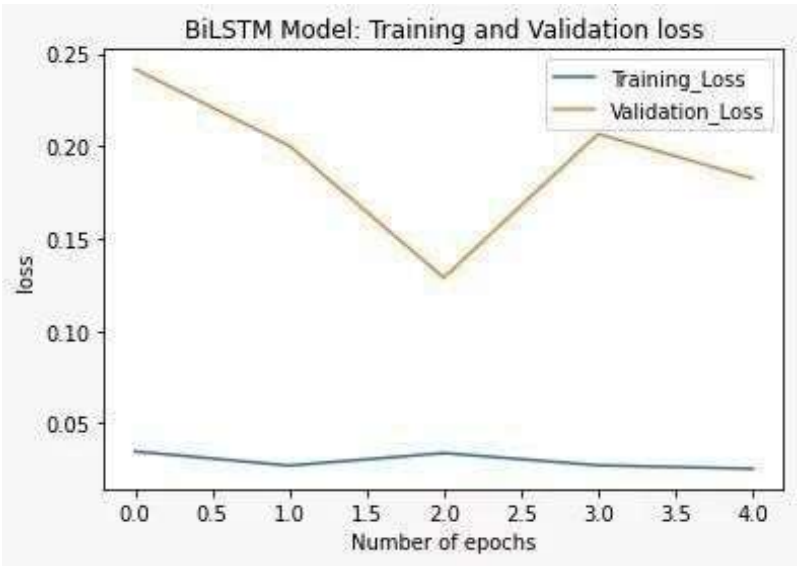
num_epochs = 30

early_stop = EarlyStopping(monitor='val_loss', patience=2) history = model2.fit(training

```
    _padded,
    train_labels,
    epochs=num_epoc
    hs,
validation_data=(testing_padded, test_labels),callbacks =[early_stop], verbose=2)
```

```
Epoch 1/30
38/38 - 0s - loss: 0.0348 - accuracy: 0.9926 - val_loss: 0.2416 - val_accuracy: 0.9492
Epoch 2/30
38/38 - 0s - loss: 0.0272 - accuracy: 0.9945 - val_loss: 0.2002 - val_accuracy: 0.9606
Epoch 3/30
38/38 - 0s - loss: 0.0339 - accuracy: 0.9923 - val_loss: 0.1288 - val_accuracy: 0.9715
Epoch 4/30
38/38 - 0s - loss: 0.0273 - accuracy: 0.9951 - val_loss: 0.2066 - val_accuracy: 0.9582
Epoch 5/30
38/38 - 0s - loss: 0.0254 - accuracy: 0.9952 - val_loss: 0.1826 - val_accuracy: 0.9629
```

And, here is our graph..



Output / Screenshots:

Get all the ham and spam messages ham_msg = messages[messages.label =='ham'] spam_msg = messages[messages.label=='spam'] # Create numpy list to visualize using wordcloud ham_msg_txt = " ".join(ham_msg.message.to_numpy().tolist()) spam_msg_txt = " ".join(spam_msg.message.to_numpy().tolist())

Since it is a text data, there are many unnecessary stopwords like articles, prepositions etc., which needs to be removed from the data.

So, let us create our wordcloud now, to extract the most frequent words in ham messages.

wordcloud of ham messages ham_msg_wcloud = WordCloud(width =520, height =260, stopwords=STOPWORDS,max_font_size=50, background_color ="red", colormap='Blues').generate(ham_msg_txt) plt.figure(figsize=(16,10)) plt.imshow(ham_msg_wcloud, interpolation='bilinear') plt.axis('off') # turn off axis plt.show()

wordcloud of spam messages

spam_msg_wcloud = WordCloud(width =520, height =260,

stopwords=STOPWORDS,max_font_size=50, background_color ="black",

colormap='Spectral_r').generate(spam_msg_txt) plt.figure(figsize=(16,10))

plt.imshow(spam_msg_wcloud, interpolation='bilinear') plt.axis('off') # turn

off axis plt.show()

The spam message WordCloud above shows that “FREE”, “call”, “text”, “GUARANTEED” and“win” are the most commonly appeared words in spam messages.

Awesome!

Now, since our data is imbalanced, we need to plot a bar graph to estimate the percentage of spam and ham ratio.

#visualize imbalanced

data

plt.figure(figsize=(8,6

))

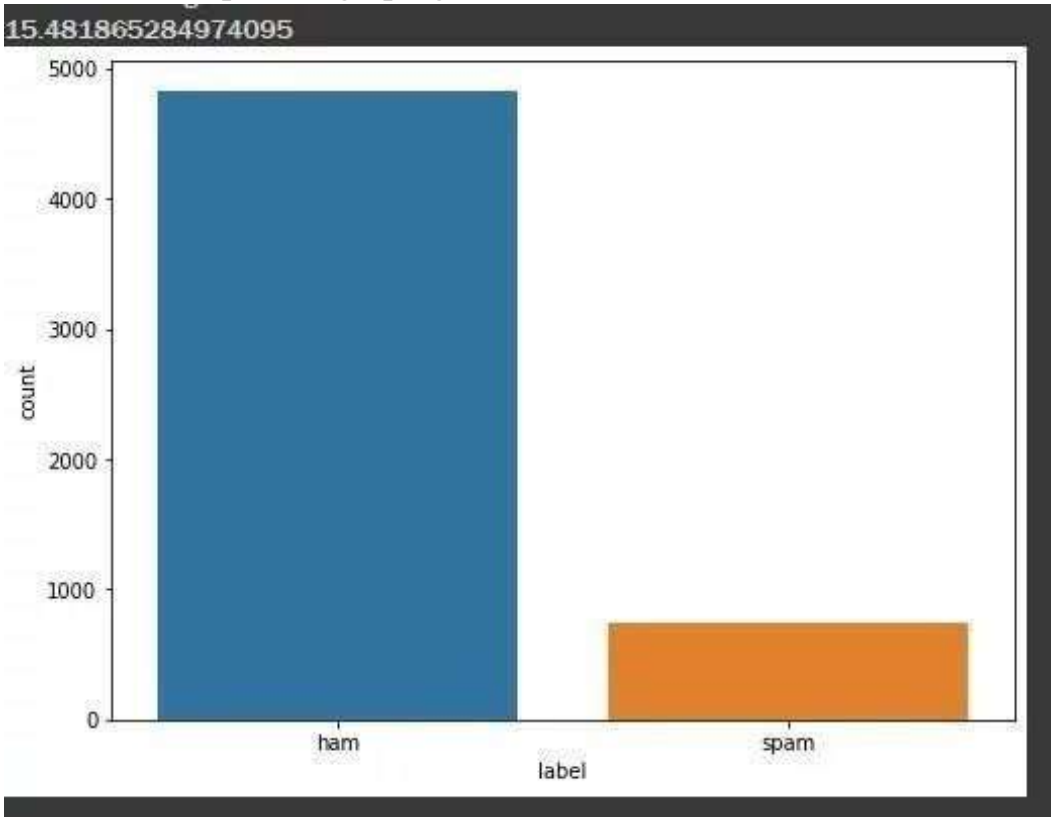
sns.countplot(messag

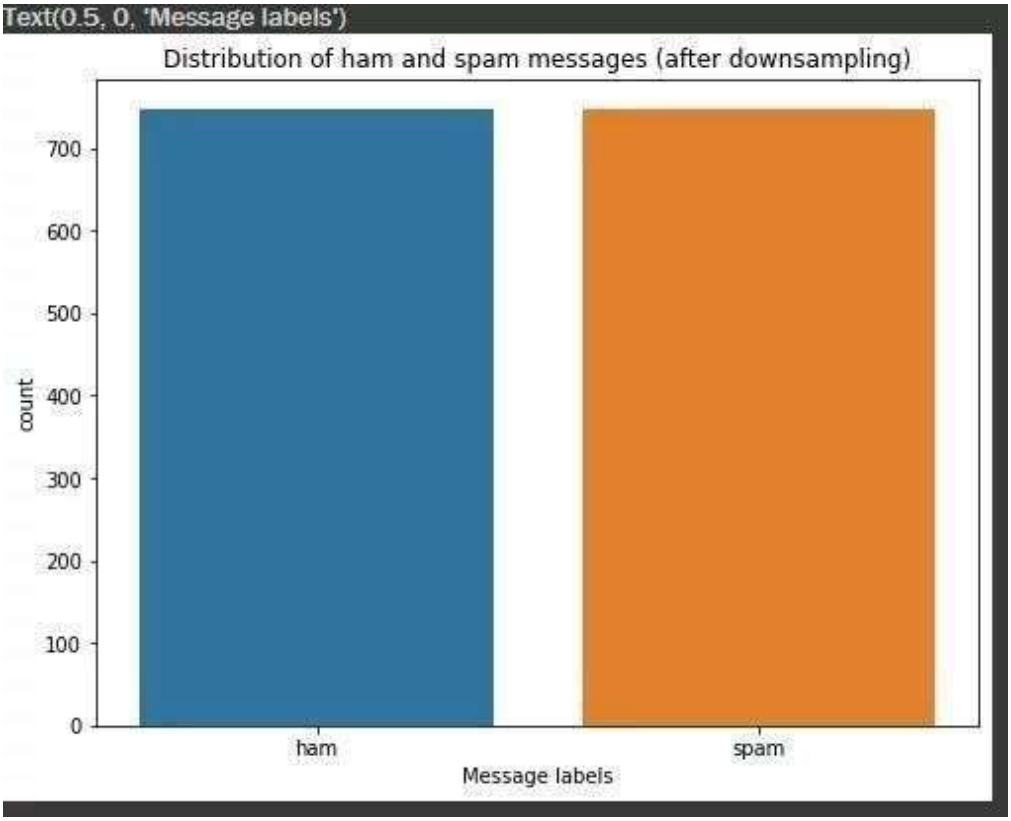
es.label) # Percentage

of spam messages

(len(spam_msg)/len(ham_msg))*100

Look at the output bar graph generated.





Get length column for each text

msg_df['text_length'] = msg_df['message'].apply(len)

#Calculate average length by label

types labels =

msg_df.groupby('label').mean()

labels

This will output us a table containing, text lengths of spam and ham messages.

	text_length
label	
ham	73.238286
spam	138.670683

Conclusion:

In conclusion, spam email detection using machine learning has become an effective and popular approach to filter out unwanted and potentially harmful emails. Various machine learning algorithms such as Support Vector Machines (SVMs), Naive Bayes, and Decision Trees have been used for this task.

The success of spam detection algorithms largely depends on the quality of the dataset used for training and testing. Feature engineering is a crucial step in the process of building an effective spam detection model. Commonly used features include the frequency of certain words, the presence of specific characters or symbols, and the overall length of the email.

Overall, spam email detection using machine learning has proven to be an effective and practical approach for filtering out unwanted emails. However, the continuous evolution of spamming techniques means that machine learning models will need to be updated regularly to maintain their effectiveness.

Future Work:

There are several areas of future work for spam email detection using machine learning:

Deep learning techniques: Deep learning techniques such as Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have shown promising results in other natural language processing tasks, and could be applied to spam email detection to improve accuracy.

Adversarial attacks: Adversarial attacks are a type of attack where malicious users attempt to evade spam detection algorithms by manipulating emails in ways that are imperceptible to human readers but can confuse machine learning algorithms. Developing spam detection algorithms that are robust to adversarial attacks is an important area of future work.

Online learning: In online learning, models are trained on a continuous stream of data and updated in real-time. This could be applied to spam detection to improve the accuracy of spam detection over time as new spamming techniques emerge.

Multilingual spam detection: Most current spam detection techniques are designed to work with English- language emails. However, as spamming techniques continue to evolve, it is important to develop multilingual spam detection models that can identify spam emails in different languages.

Privacy preservation: Spam detection systems often require access to user emails to work effectively, which can raise privacy concerns. Future work could explore ways to develop spam detection systems that preserve user privacy while still providing accurate spam detection.

Overall, the development of more accurate and robust spam detection techniques is an ongoing area of research, and there are many opportunities for future work in this field.